Introduction:

The ability to predict the stakeholders and influencers of a company's income is a powerful tool. Leveraging historical data and machine learning techniques, we aim to uncover valid insights and extrapolate using this data. Fundamentally, the challenge lies in identifying the most informative and valuable points. In this report, we aim to reveal the techniques for our decision-making.

Pre-Processing:

Upon first receiving and analyzing the data, we came to a couple of conclusions:

1. The original preprocessed data featured 106 columns of information, many with missing values and non-numerical information.
2. While some yielded a positive correlation to churn/not churning, most of the categories failed to see use.
3. While we had plenty of data, it was difficult to conclude when and why clients would opt to omit information.

With a size of over 600,000 rows and 100 columns, our priority was weeding out categories that yielded better results than others.

First, we dropped any columns with a percentage of missing data greater than 60%, because with such a high volume of customers and categories, we see it impractical to work with incomplete data at such a high frequency. Next, by manually inspecting the dataset we removed what we found to be unnecessary categorical data, including 'title', 'currency_code', 'iso_funds_code', etc.

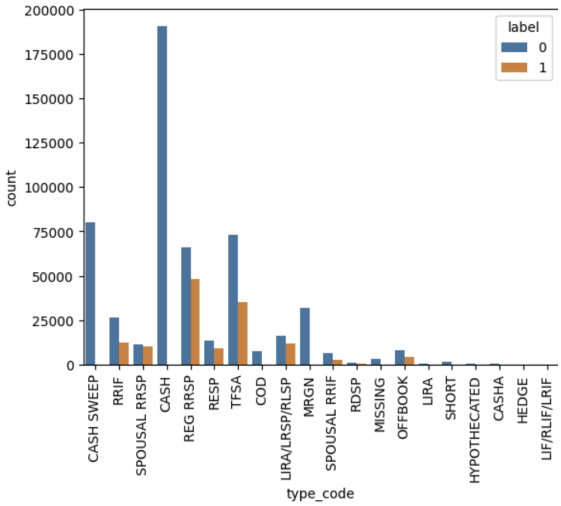
Dealing with categorical data

We used the LabelEncoder provided within the Challenge Guide to encode the test set with 1's and 0's.

The remainder of the categorical data was the first major hurdle for us. This is where we referred to online sources and the CxC workshop for how to proceed. By doing some exploratory data analysis, using the mathplotlib library to create graphs that plotted the relation between values in categorical columns and churn we found a couple key categories to consider.

These categories were: debit_code, retail_plan, and type_code. The 'debit_code' column contained 4 main values that had meaningful data in respect to churn, as a result we used a HashMap similar to the Titanic boat levels within the workshop, mapping letters to numbers. The 'retail_plan' column was encoded using one-hot encoding for simplicity. Finally, the 'type_code' column has a strong correlation with churn, for example, a customer with a CASH or MRGN account had over a 90% chance of churn. Target encoding is used in this instance over other methods because it captures the mean effect of categorical variables on the target variable; we used this because there was a strong correlation between the category and the target.

Example graph for ‘type_code’:



Finally, for categories that resulted in true or false, or two frequencies, we changed their outcome from ‘t’ and ‘f’ into 0’s and 1’s.

Before filling in the data, we dropped the IDs of the customers since they provided no valid information, similar to how the workshop dropped the passenger numbers onboard the Titanic. Additionally, we created an instance of the data frame with only numeric columns, effectively eliminating the rest of the categorical data.

Filling:

First, we discussed the type of information that needed to be included. We concluded that the type of missing data was best represented as MAR (missing at random). We believe this for a multitude of reasons:

1. Without thorough knowledge of the clients given the questions, we can’t conclude the data is MNAR (missing not at random).
2. Furthermore, assuming that the data is MCAR (missing completely at random) is too oblivious to the possibilities of results (though it would have been much more efficient to implement).
3. After researching through third-party websites, we found MAR (missing at random) is the most common and reasonable explanation for the data we did not have access to.

Ultimately, after concluding to represent our data as MAR, we used a resource linked to the project to determine the best imputation, which was multiple imputation.

Finally, after setting up the code and downloading the necessary tools, we ran the multiple imputations on our dataset and filled in our missing data.

Here, however, we ran into our first technical issue. While the resources ingrained in the notebook were great - full of detail and explanations - we still ran into some complications. One of the cons for using multiple imputations was that it would be inefficient and costly on a larger data set. Since this our first time doing a hackathon (or working with a data set entirely) we didn't know the difference between a large and small dataset.

While we wanted to use Mice imputation or regression imputation, the code would need at least over an hour to compile, and we would have to rerun the same code every time we started working on it. Realizing this was an implausible solution, we settled for using median imputation as a faster, more efficient method.

We would, however, like to stress that if there wasn't an issue or we had more resources at hand to implement a stronger technique, we would vouch for using MICE imputation or regression imputation to ultimately yield stronger results.

Now our data was finally fully processed and ready to be tested.

Testing:

We chose XGBoost over the Decision Tree Classifier because it is inherently more suited for handling imbalanced datasets, where the target events or classes might be underrepresented(in our case this was churn outweighing no-churn). Additionally, its built-in regularization helps prevent overfitting, which is crucial when dealing with sparse data. These features made XGBoost our classifier of choice since it is not biased towards the majority class.

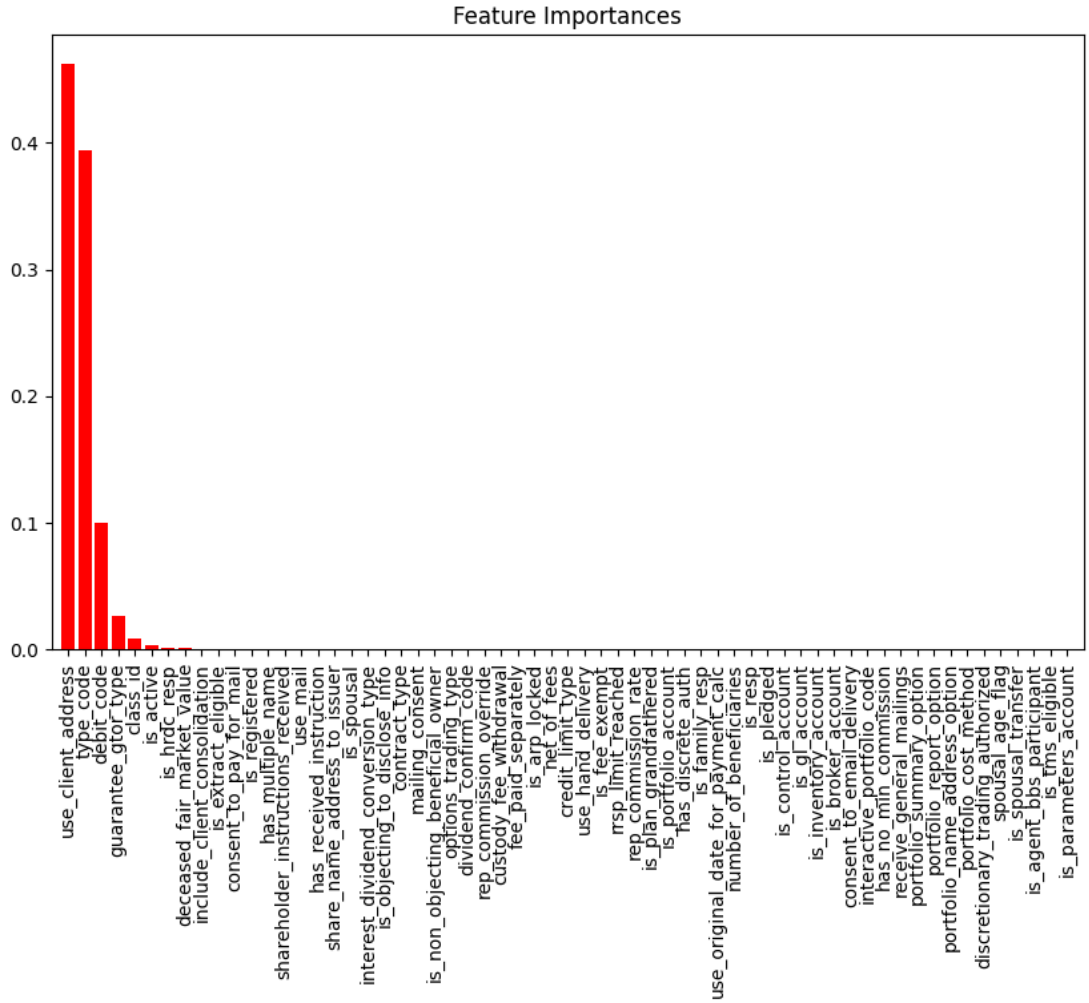
Hyperparameter Tuning

In our project, we implemented hyperparameter tuning to optimize the performance scores of our predictive model using RandomizedSearchCV. This method was preferred over exhaustive grid search methods due to its computational efficiency and faster execution time, which was particularly advantageous when working with large datasets and complex models like XGBoost. By specifying a range of potential values for key hyperparameters such as the number of estimators, learning rate, and tree depth, RandomizedSearchCV evaluated different models using 3-fold cross-validation to assess their performance. We focused on maximizing the accuracy score, but this process can be easily adapted to optimize other metrics like the F1 score by adjusting the scoring parameter. The randomized nature of the search allowed us to cover a broad range of the hyperparameter space, increasing the likelihood of finding a highly effective model configuration while significantly reducing the computational burden typically associated with hyperparameter optimization.

By implementing these processes we produced strong performance on our evaluation metrics:

1. **F1 Score:** 0.9242610837438423
2. **MCC Score:** 0.9070376440751509
3. **Average CV Score:** 0.9656395097286417

Marketing Strategies / Feature Analysis:



To investigate and develop a marketing strategy, we opted to use a feature analysis to determine the most influential categories so we were more informed with our marketing decisions. After running a feature analysis, we observed that client address, type of account, and debit code were the majority of stakeholders in predicting churn or no churn.

From this conclusion, we hypothesized that the type of account and service was directly correlated to the longevity of a client - a cash account would yield a higher churn percentage, while a longer-standing

type of account would see less churning. Extrapolating from this data, we propose 2 marketing strategies:

1. **Incentives:**

To push clients towards longer-standing accounts, we want there to be more benefits, programs, and loyalties attached to these accounts. For example, after a specified number of years (every 2 years), we offer a discounted rate, waived fees, and other perks such as higher tiered interest rates that increase over time. We believe that this will prompt more people to purchase longer-standing accounts which directly correlates to a client base with less churning.

2. **Maintenance**

To keep long-standing and current clients, we will offer a relationship pricing model that encourages and delivers more rewards for the longer time spent with the company. On top of this, we encourage community involvement and bonds to ensure that our clients feel important and more included. Developing a relationship with our clients further cements our loyalty and highlights our commitment. An example of community involvement would be

- Financial education programs as well as financial counseling, where we organize workshops and collaborate with schools.
- Donations and sponsorships to community events like festivals or other community events.

Conclusion / Reflection:

Knowledge has never been more powerful, both as a tool and a resource. The ML-processed data yielded extremely high results, and given more access to technology and experience, we expect it could be even greater.

In our time dedicated to this hackathon, nearly 50% was spent on readings, studying the tools and techniques in processing datasets, and observing the correlation between a category and the final result. We would love the chance to do this again, now with more experience and understanding. While our code was sufficient for this challenge, we realized there is much more to learn and understand in managing data.

Next Steps:

For the next hackathon, we know to focus primarily on results-oriented code. Something we noticed throughout our time working together was that we dedicated plenty of time, thought, and effort to ideas and code that were ultimately meaningless or ineffective. Before starting on an idea, we should have a better grasp of knowledge and command of the coding language. Furthermore, neither of us was able to attend the in-person seminar on account of conflicting schedules, so both of us had to rewatch and learn the content on our own time through a video, which we felt was less than ideal.